Heterogeneous Modeling of Embedded Software

Workshop on New Visions for Software Design and Productivity

SDP 2001

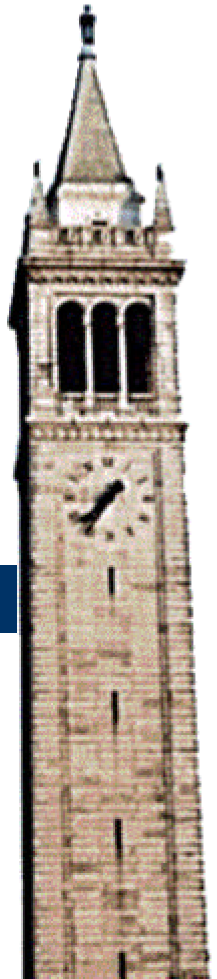
Nashville, Tennessee

13-14 December, 2001

H John Reekie

Edward Lee

UC Berkeley

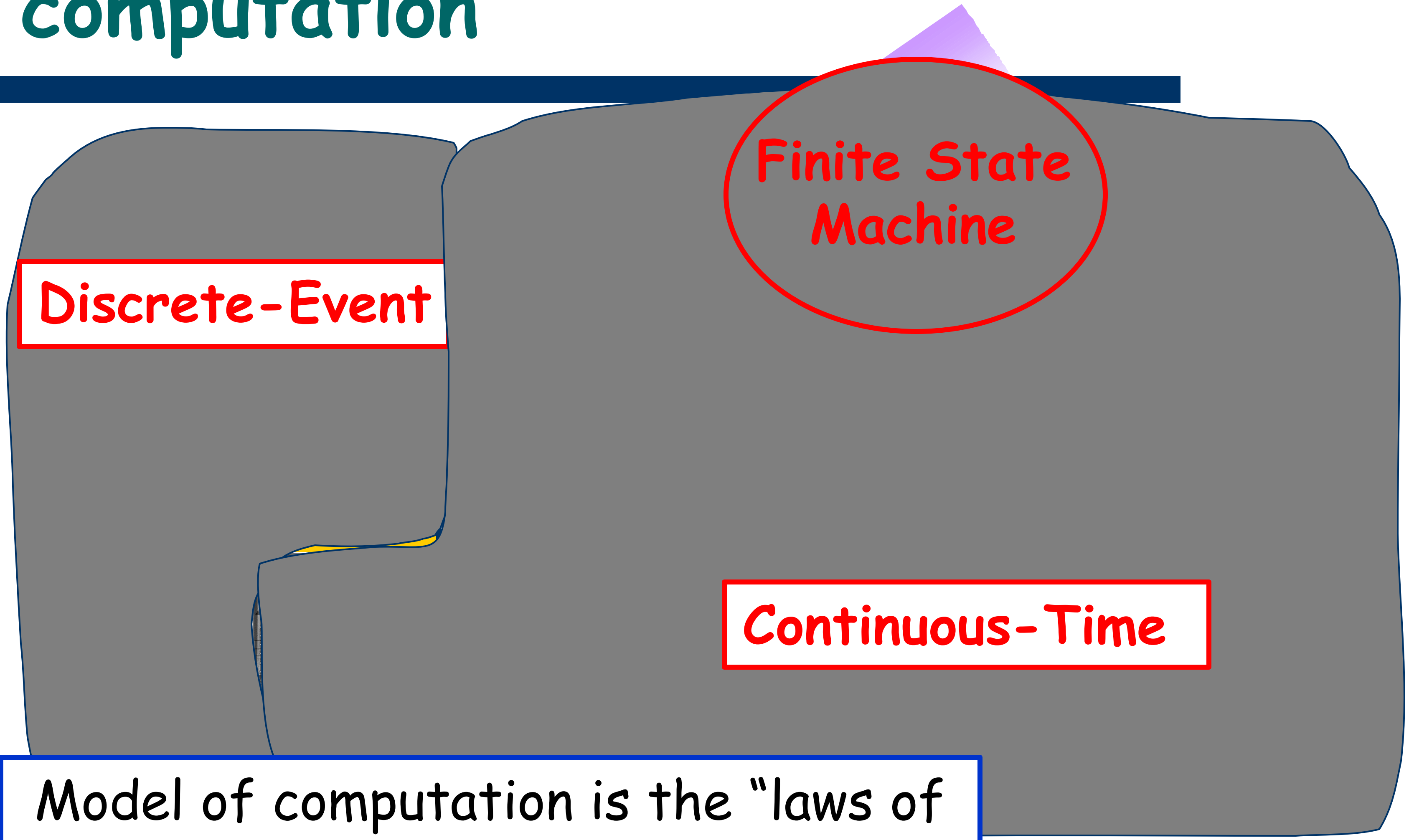


Aspects of embedded software

- Interaction with physical processes
  - sensors, actuators, processes
- Critical properties are not all functional
  - real-time, fault recovery, power, security, robustness
- Heterogeneous
  - hardware/software, mixed architectures
- Concurrent
  - interaction with multiple processes
- Reactive
  - operating at the speed of the environment

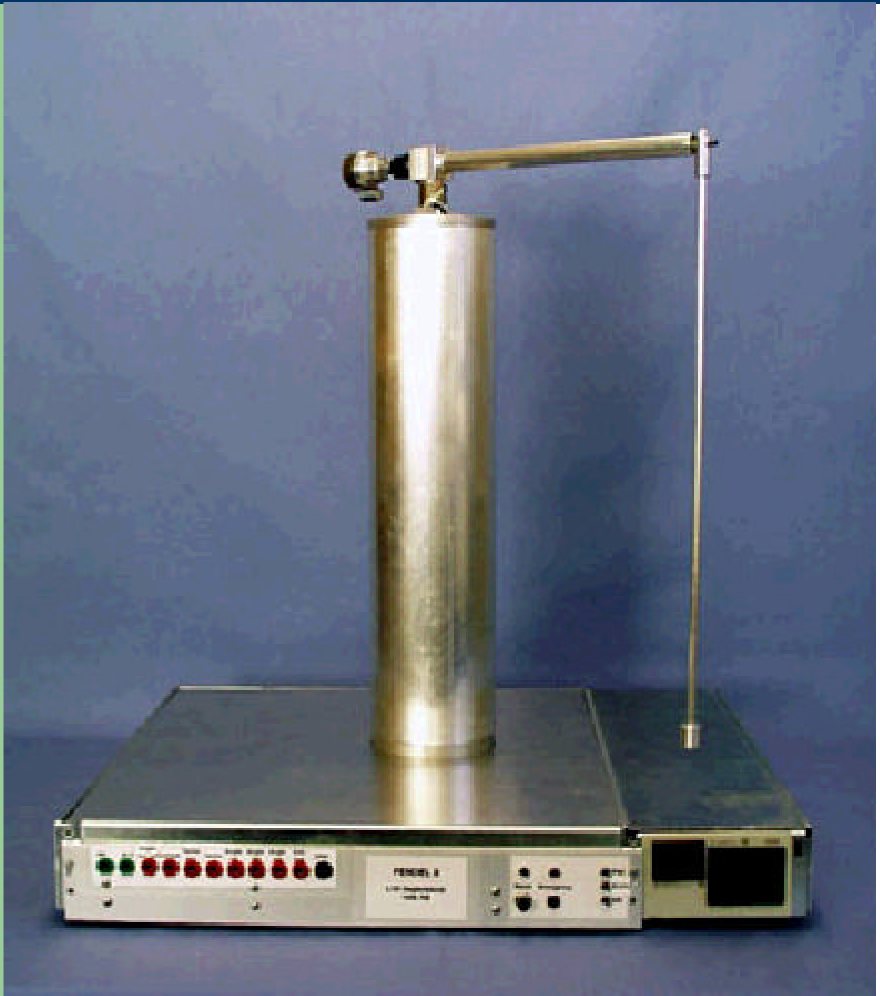
These features look more like hardware!

Heterogeneous models of computation



Model of computation is the "laws of physics" of component interaction

Example: Controlling an inverted pendulum

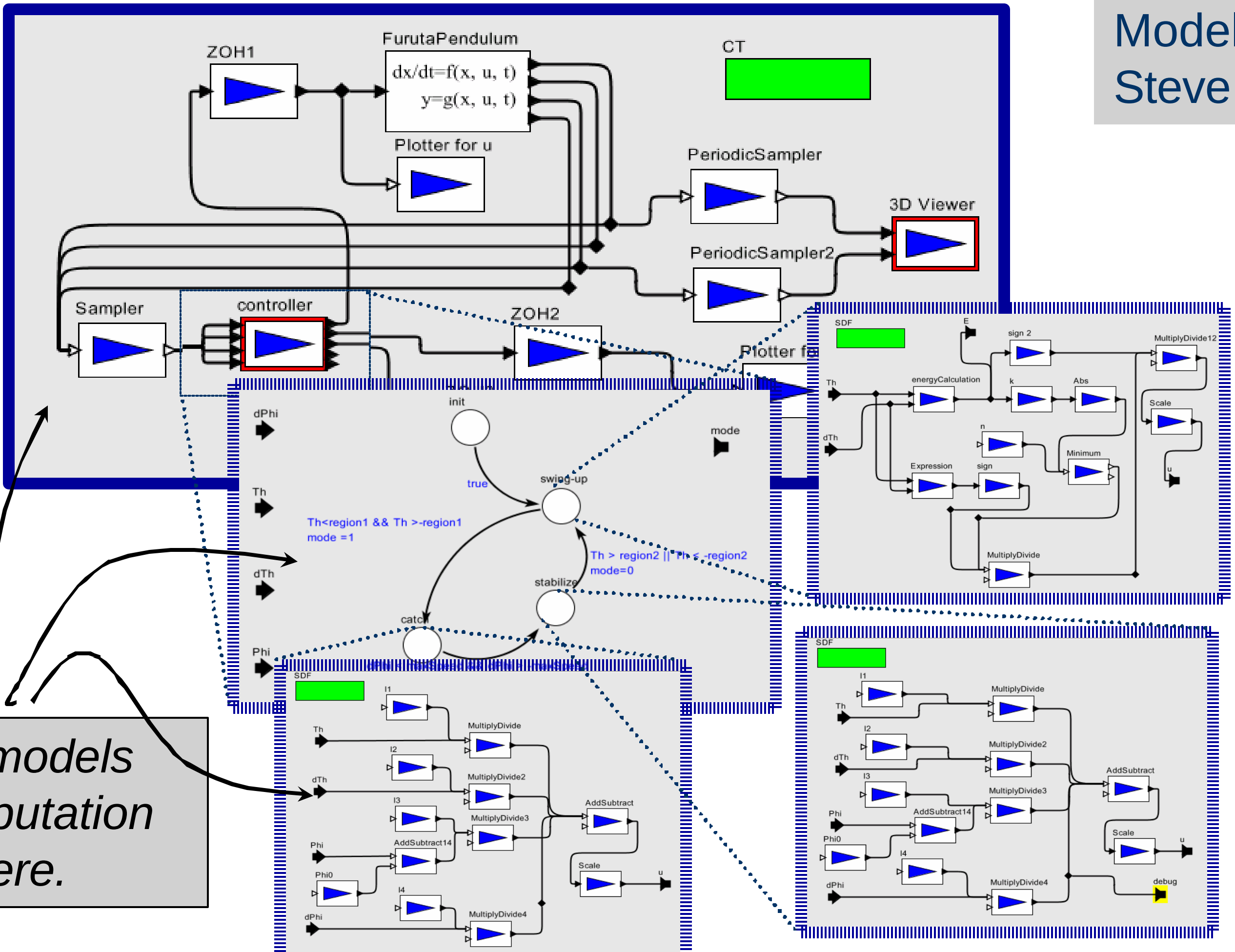


The Furuta pendulum has a motor controlling the angle of an arm, from which a free-swinging pendulum hangs. The objective is to swing the pendulum up and then balance it.

Representative of many embedded systems

Hierarchical Heterogeneity

Models by Jie Liu and Steve Neuendorffer

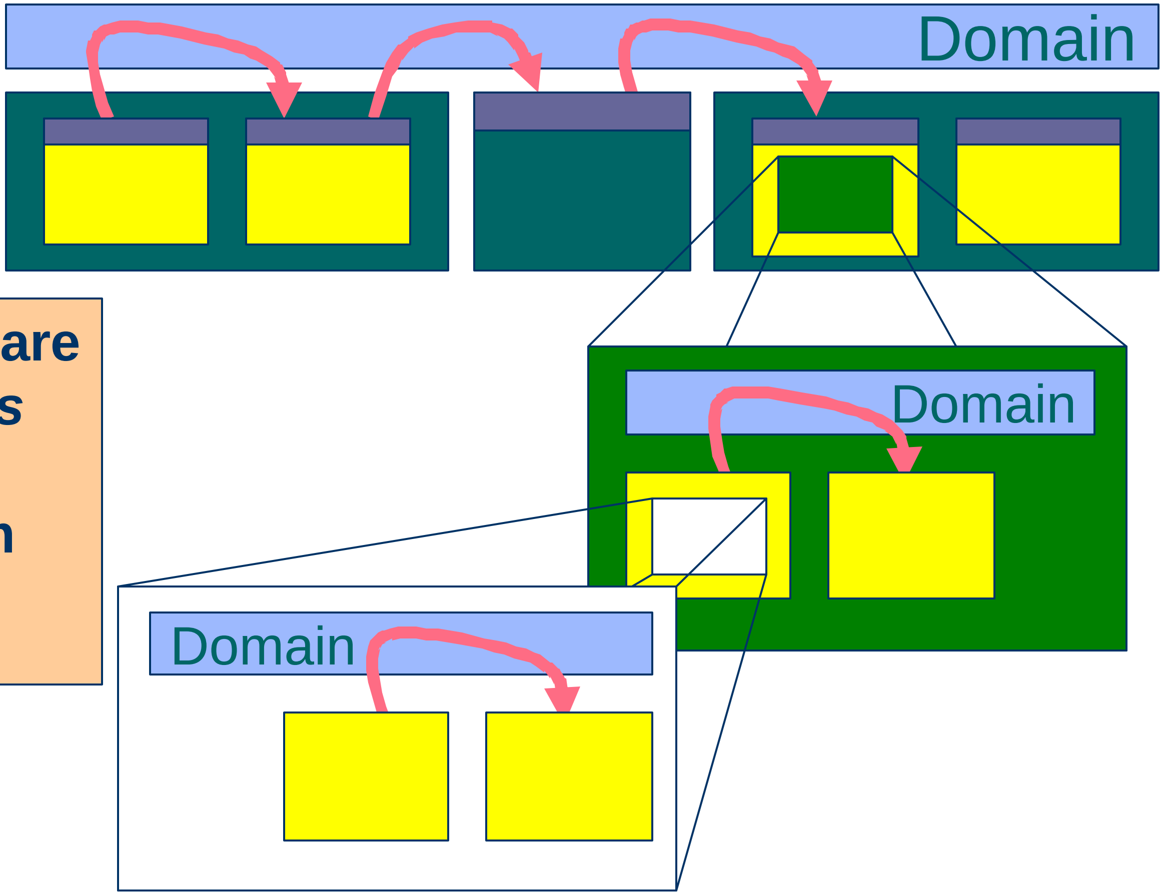


Components are actors with ports

Model of computation controls interaction

Three models of computation used here.

Therefore: Hierarchical, Compositional Models are Key



Actors with ports **are better than *objects with methods* for embedded system design.**

A Laboratory for Exploring Component Frameworks

Ptolemy II –

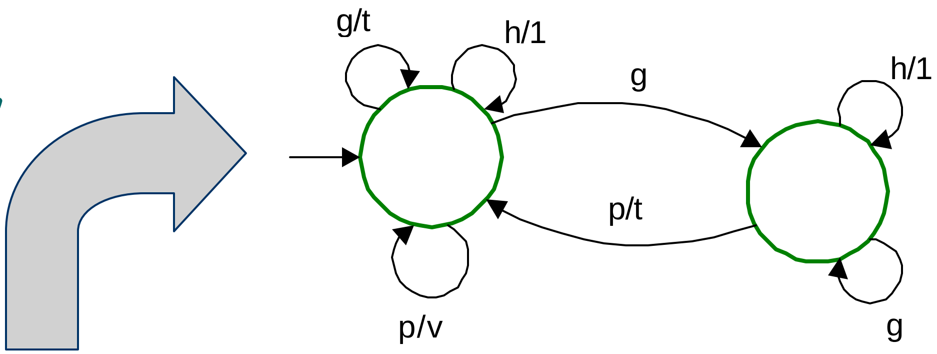
- Java based, network integrated
- Several frameworks implemented

A realization of a model of computation is called a "domain." Multiple domains can be mixed hierarchically in the same model.

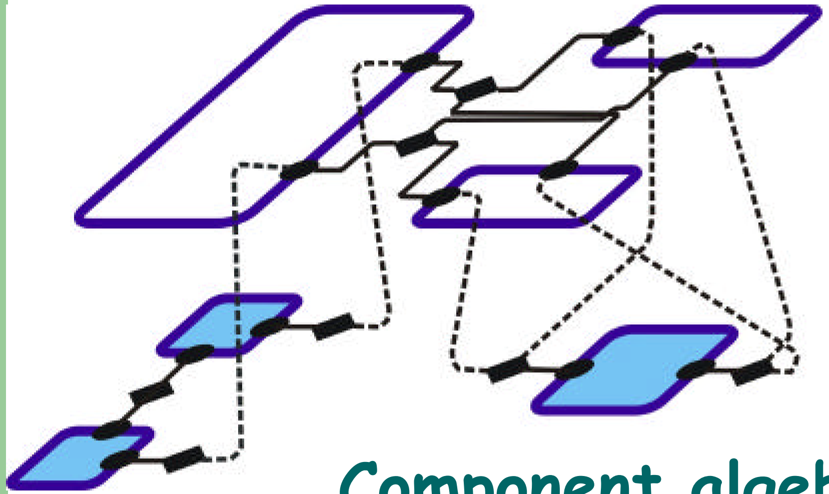
http://ptolemy.eecs.berkeley.edu

Interface Theories (de Alfaro and Henzinger)

"Implements"



Interface algebra

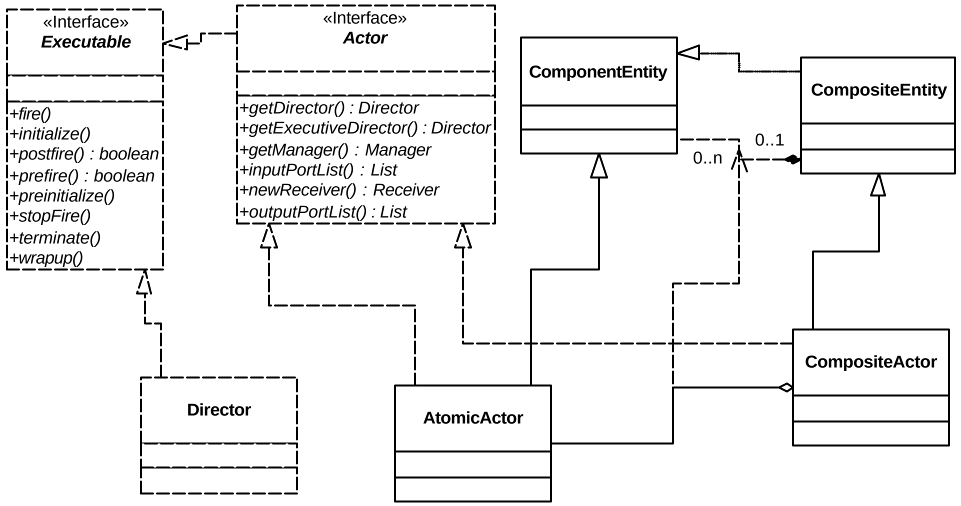


Component algebra

- Functional requirements
- Timing constraints
- Liveness and concurrency
- Refinement

Implementation Architecture - API

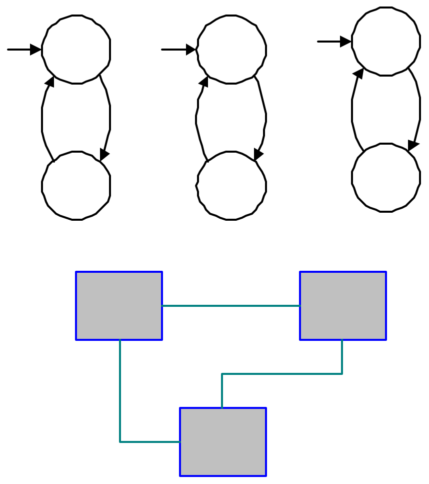
- Programmer's API exposes component model and an execution model
  - Conventional, well-understood
  - Difficult to extend, single-language



Ptolemy

Implementation Architecture – Compile to Abstract Machine

- Separates programmer's model from implementation model
  - Extensible, retargetable, optimizable
  - Supports "real" embedded systems



component

channel

event

time

action

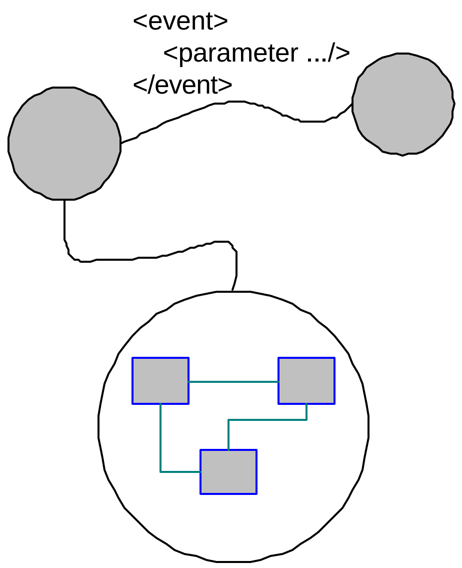
transition

… ? …

e-machine
Calif

Implementation Architecture - Protocol

- Simple protocol exposes MoC “primitives”
  - Distributed, cross-language, legacy support
  - Clients, servers, peers



tagged sequences

precise reaction

… ? …

eg Nephest?

Conclusions

- Software experts are unlikely to solve the embedded software problem on their own.
- Actors with ports are better than objects with methods for embedded system design.
- Well-founded models of computation matter a great deal.
- Further research can extend the application of hierarchical heterogenous models of computation in embedded systems.